

```

%EN ESTE LA FUNCION OBJETIVO CONSIDERA TIEMPO DE ESPERA Y CANTIDAD DE
%PERSONAL
clc

clear all
close all
Tr = 60;
[I,CCP_h,CTE_i,M,TH_hj,R_cj,CP_ic,TP_ij,L_i,TM_c,epsilon,D_ij,AP_ijh] = PARAMETROS5();
    %Numero grande;
D2_ij = D_ij;
for i = 1:I
    for j = 1:12
        if D2_ij(i,j) == 0
            D2_ij(i,j) = 1000000;
        end
    end
end
%IC_i=[0.06 0.03 0.06];
%Variables
S_ijpk = optimvar('S_ijpk',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2)],↵
    'Type', 'integer','LowerBound',0, 'UpperBound',1);%1 Si ij va antes que pk
T_ijpk = optimvar('T_ijpk',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2)],↵
    'Type', 'integer','LowerBound',0, 'UpperBound',1);%1 Si ij va antes que pk
R_ijpk = optimvar('R_ijpk',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2)],↵
    'Type', 'integer','LowerBound',0, 'UpperBound',1);%1 Si ij inician desues de pk
B_ijpk = optimvar('B_ijpk',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2)],↵
    'Type', 'integer','LowerBound',0, 'UpperBound',1);%Desactivar recurso interno
U_ijpk = optimvar('U_ijpk',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2)],↵
    'Type', 'integer','LowerBound',0, 'UpperBound',1);%Desactivar recurso interno
W_ijpk = optimvar('W_ijpk',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2)],↵
    'Type', 'integer','LowerBound',0, 'UpperBound',1);%Desactivar recurso interno
Q_ijpkh = optimvar('Q_ijpkh',[size(TP_ij,1), size(R_cj,2),size(TP_ij,1), size(R_cj,2),↵
    size(TH_hj,1)], 'Type', 'integer','LowerBound',0, 'UpperBound',1);%1 Si ij va antes↵
    que pk
TI_ij = optimvar('TI_ij',[size(TP_ij,1), size(TH_hj,2)], 'Type',↵
    'continuous','LowerBound', 0); %Momento de inicio de tarea
TE_ij = optimvar('TE_ij',[size(TP_ij,1), size(TH_hj,2)], 'Type',↵
    'continuous','LowerBound', 0); %Tiempo de espera antes de la tarea j
TF_ij = optimvar('TF_ij',[size(TP_ij,1), size(TH_hj,2)], 'Type',↵
    'continuous','LowerBound', 0); %Tiempo de finalizacion de la tarea j del paciente i
HT_h = optimvar('HT_h',[1 4], 'Type', 'integer','LowerBound',1,'UpperBound',I);
%Función objetivo
Z = optimproblem("ObjectiveSense","min");
Z.Objective = sum(CCP_h.*HT_h,"all") + sum(CTE_i*sum((TE_ij.*D2_ij)/10,2),1);
%Z.Objective = sum(CCP_h.*HT_h,"all") + sum(CTE_i.*(TE_ij/10),"all");

tstart = tic;
%RESTRICCION 1-----
for i = 1:size(CP_ic,1);
    for j= 2:size(TH_hj,2);
        Z.Constraints.(['R1_i',num2str(i),'_j',num2str(j)]) = ...
            TE_ij(i,j) >= TI_ij(i,j) - (TI_ij(i,j-1) + TP_ij(i,j-1).*D_ij(i,j-1));
    end
end
end

```

```

disp('R1')
%RESTRICCION 2-----
for i = 1:size(CP_ic,1)
    Z.Constraints.(['R2_i',num2str(i)]) = ...
    TI_ij(i,1) >= L_i(i);
end
disp('R2')
%RESTRICCION 3-----
% Z.Constraints.(['R3']) = TI_ij(1,1) == L_i(1);
% disp('R3')
%RESTRICCION 4-----
for i = 1:size(CP_ic,1);
    for j = 2:size(TH_hj,2);
        Z.Constraints.(['R4_i',num2str(i),'_j',num2str(j)]) = ...
        TI_ij(i,j) >= TF_ij(i,j-1);
    end
end
disp('R4')
%RESTRICCION 5-----
for i = 1:size(CP_ic,1);
    Z.Constraints.(['R5_i',num2str(i)]) = ...
    TE_ij(i,1) == TI_ij(i,1) - L_i(i);
end
disp('R5')
%RESTRICCION 6-----
for i = 1:size(CP_ic,1);
    for j = 1:size(TH_hj,2);
        Z.Constraints.(['R6_i',num2str(i),'_j',num2str(j)]) = ...
        TF_ij(i,j) == TI_ij(i,j) + TP_ij(i,j).* D_ij(i,j);
    end
end
disp('R6')
%RESTRICCION 7-----
[N_pacientes, ~] = size(CP_ic);
[p_idx, i_idx] = find(~eye(N_pacientes)); % Todas las combinaciones p ≠ i
num_pares = length(p_idx);
N_tareas = size(TH_hj, 2);
N_recursos = size(TH_hj, 1);
restricciones_R7 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_recursos
            restricciones_R7(contador) = M * R_ijpk(i,j,p,k) >= TF_ij(i,j) - TI_ij(p,
k);
            contador = contador + 1;
        end
    end
end
Z.Constraints.R7 = restricciones_R7;
disp('R7 optimizado');
%R7.1
restricciones_R7_1 = optimconstr(num_pares * N_tareas^2);

```

```

contador = 1;
for idx = 1:num_pares
    i = i_idx(idx);
    p = p_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R7_1(contador) = M*(1-R_ijk(i,j,p,k)) >= (TI_ij(p,k) - ✓
TF_ij(i,j)) + epsilon;
            contador = contador + 1;
        end
    end
end
Z.Constraints.R7_1 = restricciones_R7_1;
disp('R7.1 optimizado');
%RESTRICCION 8-----
restricciones_R8 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R8(contador) = M * S_ijk(i,j,p,k) >= TI_ij(i,j) - TI_ij(p, ✓
k);
            contador = contador + 1;
        end
    end
end
Z.Constraints.R8 = restricciones_R8;
disp('R8 optimizado');
% Restricción R8.1 optimizada
restricciones_R8_1 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R8_1(contador) = M * (1 - S_ijk(i,j,p,k)) >= (TI_ij(p,k) - ✓
TI_ij(i,j)) + epsilon;
            contador = contador + 1;
        end
    end
end
Z.Constraints.R8_1 = restricciones_R8_1;
disp('R8.1 optimizado');
%RESTRICCION 9-----
restricciones_R9 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R9(contador) = M*(U_ijk(i,j,p,k)) >= TF_ij(i,j) - TF_ij(p, ✓

```

```

k);
        contador = contador + 1;
    end
end
end
Z.Constraints.R9 = restricciones_R9;
disp('R9 optimizado');
%R9.1
restricciones_R9_1 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R9_1(contador) = M*(1 - U_ijk(i,j,p,k)) >= (TF_ij(p,k) -
TF_ij(i,j)) + epsilon;
            contador = contador + 1;
        end
    end
end
Z.Constraints.R9_1 = restricciones_R9_1;
disp('R9.1 optimizado');
%RESTRICCION 10-----
restricciones_R10 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R10(contador) = M*(W_ijk(i,j,p,k)) >= TI_ij(i,j) - TF_ij(p,
k);
            contador = contador + 1;
        end
    end
end
Z.Constraints.R10 = restricciones_R10;
disp('R10 optimizado');
%R10.1
restricciones_R10_1 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R10_1(contador) = M*(1 - W_ijk(i,j,p,k)) >= (TF_ij(p,k) -
TI_ij(i,j)) + epsilon;
            contador = contador + 1;
        end
    end
end
Z.Constraints.R10_1 = restricciones_R10_1;
disp('R10.1 optimizado');

```

```

%RESTRICCION 11-----
restricciones_R11 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R11(contador) = 5*(B_ijpk(i,j,p,k)) >= ((R_ijpk(i,j,p,k) - S_ijpk(i,j,p,k)) + (U_ijpk(i,j,p,k) - W_ijpk(i,j,p,k))) - 1;
            contador = contador + 1;
        end
    end
end
Z.Constraints.R11 = restricciones_R11;
disp('R11 optimizado');
%R11.1
restricciones_R11_1 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R11_1(contador) = 5*(1 - B_ijpk(i,j,p,k)) >= 1 - ((R_ijpk(i,j,p,k) - S_ijpk(i,j,p,k)) + (U_ijpk(i,j,p,k) - W_ijpk(i,j,p,k))) + epsilon;
            contador = contador + 1;
        end
    end
end
Z.Constraints.R11_1 = restricciones_R11_1;
disp('R11_1 optimizado');
%RESTRICCION 12-----
restricciones_R12 = optimconstr(num_pares * N_tareas^2);
contador = 1;
for idx = 1:num_pares
    p = p_idx(idx);
    i = i_idx(idx);
    for j = 1:N_tareas
        for k = 1:N_tareas
            restricciones_R12(contador) = T_ijpk(i,j,p,k) == R_ijpk(i,j,p,k) - S_ijpk(i,j,p,k) + U_ijpk(i,j,p,k) - W_ijpk(i,j,p,k) - B_ijpk(i,j,p,k);
            contador = contador + 1;
        end
    end
end
Z.Constraints.R12 = restricciones_R12;
disp('R12 optimizado');
%RESTRICCION 13-----
restricciones_R13 = optimconstr(N_pacientes, N_tareas, 4); % [i, j, h]
for h = 1:4
    for i = 1:N_pacientes
        p_validos = p_idx(i_idx == i); % Índices p donde p ≠ i
        num_p = length(p_validos); % Número de pares válidos
        for j = 1:N_tareas

```

```

        k_values = 2:N_tareas;
        AP_sub = AP_ijh(p_validos, k_values-1, h); % Dimensión: [num_p x (N_tareas-1)]
        Q_sub = squeeze(Q_ijpkh(i, j, p_validos, k_values-1, h)); % Dimensión: [num_p x (N_tareas-1)]
        sum1 = sum(AP_sub .* Q_sub, 'all');
        AP_sub2 = AP_ijh(p_validos, 12, h); % Dimensión: [num_p x 1]
        T_sub = squeeze(T_ijpk(i, j, p_validos, 12)); % Dimensión: [num_p x 1]
        sum2 = sum(AP_sub2 .* T_sub, 'all');
        restricciones_R13(i, j, h) = AP_ijh(i, j, h) + sum1 + sum2 <= HT_h(h);
    end
end
end
Z.Constraints.R13 = restricciones_R13(:);
disp('R13 optimizado corregido');
%RESTRICCION 14-----
restricciones_R14 = optimconstr(num_pares * (N_tareas - 1) * N_recursos * N_tareas);
contador = 1;
for idx = 1:num_pares
    i = i_idx(idx);
    p = p_idx(idx);
    for j = 1:N_tareas
        for k = 2:N_tareas
            t_values = k:N_tareas;
            for h = 1:N_recursos
                AP_sub = squeeze(AP_ijh(p, t_values, h)); % Dimensión: [1 x length(t_values)]
                T_sub = squeeze(T_ijpk(i, j, p, t_values)); % Transponer a [1 x length(t_values)]
                sumAP_T = sum(AP_sub .* T_sub, 2);
                restricciones_R14(contador) = Q_ijpkh(i, j, p, k-1, h) >= T_ijpk(i, j, p, k-1) - sumAP_T;
                contador = contador + 1;
            end
        end
    end
end
Z.Constraints.R14 = restricciones_R14;
disp('R14 optimizado');
%R14.1
restricciones_R14_1 = optimconstr(num_pares * (N_tareas - 1) * N_recursos * N_tareas);
contador = 1;
for idx = 1:num_pares
    i = i_idx(idx);
    p = p_idx(idx);
    for j = 1:N_tareas
        for k = 2:N_tareas
            t_values = k:N_tareas;
            for h = 1:N_recursos
                AP_sub = squeeze(AP_ijh(p, t_values, h)); % Dimensión: [1 x length(t_values)]
                T_sub = squeeze(T_ijpk(i, j, p, t_values)); % Transponer a [1 x length(t_values)]
                sumAP_T = sum(AP_sub .* T_sub, 2);

```

```

        restricciones_R14_1(contador) = M*(1 - Q_ijpkh(i,j,p,k-1,h)) >= ✓
sumAP_T - T_ijpk(i,j,p,k-1) + epsilon;
        contador = contador + 1;
    end
end
end
end

Z.Constraints.R14_1 = restricciones_R14_1;
disp('R14.1 optimizado');
%RESTRICCION 15-----
for i = 1:size(CP_ic,1);
    sumTE = sum(TE_ij(i,:));
    Z.Constraints.(['R15_i',num2str(i)]) = ...
    sumTE <= Tr;
end
disp('R15')
% %RESTRICCION 16-----
for i = 1:size(CP_ic,1);
    sumTE = sum(TE_ij(i,1:4));
    Z.Constraints.(['R16_i',num2str(i)]) = ...
    CP_ic(i,1).* sumTE <= TM_c(1).*CP_ic(i,1);
end
disp('R16')
% %RESTRICCION 17-----
for i = 1:size(CP_ic,1);
    sumTE = sum(TE_ij(i,1:4));
    Z.Constraints.(['R17_i',num2str(i)]) = ...
    CP_ic(i,2).* sumTE <= TM_c(2).*CP_ic(i,2);
end
disp('R17')
% %RESTRICCION 18-----
for i = 1:size(CP_ic,1);
    for c = 3:5;
        sumTE = sum(TE_ij(i,1:4));
        Z.Constraints.(['R18_i',num2str(i),'_c',num2str(c)]) = ...
        CP_ic(i,c).* sumTE <= TM_c(c).*CP_ic(i,c);
    end
end
disp('R18')
tend = toc(tstart)

sol = solve(Z, 'Options', optimoptions('intlinprog', ...
    'AbsoluteGapTolerance', 1000, 'RelativeGapTolerance', 1e-02, ...
    'ConstraintTolerance', 1e-04, 'LPOptimalityTolerance', 1e-4, 'Display', ...
    'iter', 'CutGeneration', 'advanced', 'Heuristics', 'advanced', 'MaxTime', 14400));

% disp('Solución de TI(i,j): ')
% disp(sol.TI_ij)
% disp('Solución de TF(i,j): ')
% disp(sol.TF_ij)
disp('Solución de TE(i,j): ')
disp(sol.TE_ij)
disp('La cantidad de personal HT(h): ')

```

```
disp(sol.HT_h)
```

```
%Grafica-----
TI = sol.TI_ij;
TF = sol.TF_ij;
TE = sol.TE_ij;
num_pacientes = size(TI,1);
num_actividades = size(TI,2);
figure('units','normalized','outerposition',[0 0 1 1],'Color',[0.5 0.5 0.5]);
ax = gca;
set(ax, 'Color', [0.5 0.5 0.5]);
grid on;
hold on;
colores = lines(num_pacientes);
for i = 1:num_pacientes
    for j = 1:num_actividades
        inicio = TI(i, j);
        duracion = TF(i,j) - TI(i,j);
        rectangle('Position', [inicio, i-0.2, duracion, 0.4], 'FaceColor', colores(i,
:), 'EdgeColor', 'k');
    end
end
xlabel('Tiempo');
ylabel('Paciente');
yticks(1:num_pacientes);
yticklabels(arrayfun(@(x) ['Paciente ' num2str(x)], 1:num_pacientes, 'UniformOutput',
false));
xticks(0:1:1000);
title('Diagrama de Gantt de Pacientes y Actividades');
hold off;
```